

```

File: puppetrepo-shoal/files/set-session-proxy.sh
proxy="/usr/bin/shoal-client -d`
#echo $proxy
if [[ $proxy =~ 'export http_proxy=http://' ]];
then
$proxy > /usr/run-export-command.sh
chmod 755 /usr/run-export-command.sh
source /usr/run-export-command.sh
echo "Proxy set successfully!"
else
echo "Proxy not set!"
echo "No squid servers are active currently!"
fi
-----
-----
-----

```

```

File: puppetrepo-shoal/files/shoal-client
#!/usr/bin/python
"""
    Very simple client script used to get nearest squid server using the RESTful API.
"""
import urllib2
import sys
# import json
import re
import os
import logging
import time

from shoal_client import config as config

```

```

from optparse import OptionParser
from urllib2 import urlopen

server = config.shoal_server_url
cvdfs_config = config.cvdfs_config
default_http_proxy = config.default_squid_proxy

data = None
dump = False
closest_http_proxy = ''
http_proxy_formatted = ''
cvdfs_http_proxy = "CVMFS_HTTP_PROXY="

logging.basicConfig(filename="shoal_client.log", format='%(asctime)s %(message)s')

def get_args():
    """
        gets server and dump variables from command line arguments
    """
    global server
    global dump

    p = OptionParser()

```

```

p.add_option("-s", "--server", action="store", type="string", dest="server",
             help="Also needs string for specifying the shoal server to use. " +
             "Takes presedence over the option in config file")
p.add_option("-d", "--dump", action="store_true", dest="dump",
             help="Print closest proxies to terminal for debugging "+
             "instead of over writing the CVMFS config file")
(options, args) = p.parse_args()

if options.server:
    server = options.server
if options.dump:
    dump = True

def convertServerData(val):
    """
    converts val to digits if it's not already or else return None
    """
    if val.isdigit():
        return int(val)
    else:
        try:
            return float(val)
        except:
            if "null" in val:
                return None
            else:
                return unicode(val.strip("\\"))

# TODO is this parser sufficient or should a full JSON parser be implemented?
# Seperating out the list of properties should be done but does support
# for arbitrary json strings add anything?
def parseServerData(jsonStr):
    """
    creates a multidimensional server data dictionary indexed by
    unicode integers with dataTypes, geo_data and geoDataTypes. Each
    respective entry holds the appropriate dataTypes and geoDataTypes
    found in jsonStr
    """

    # TODO should load this from a config file as it has to match the server
    # Nested properties (i.e geo_data) needs to be handled separately
    dataTypes = ["load", "distance", "squid_port", "last_active", "created",
                 "external_ip", "hostname", "public_ip", "private_ip"]

    geoDataTypes = ["city", "region_name", "area_code", "time_zone", "dma_code",
                    "metro_code", "country_code3", "latitude", "postal_code",
                    "longititude", "country_code", "country_name", "continent"]

    # don't really care about data here

```

```

# it is just a simple way to get number of nearest squids
p = re.compile("\\" + dataTypes[0] + "\": ([^,]+)")
numNearestSquids = len(p.findall(jsonStr))
## compiles regex "load": ([^,]+), although it doesn't really matter that fact that it's a load
## this will find the number of above matches in jsonStr and return into numNearestSquids
## therefore each match in json is a 'nearest' squid

# initialize the dictionaries
outDict = {}
for i in range(0, numNearestSquids):
    outDict[unicode(str(i))] = {}
    outDict[unicode(str(i))][unicode("geo_data")] = {}
## creates a multidimensional dict with each key 1 being u'i' (i in unicode)
## and key 2 being "geo_data" for all entries

# TODO probably don't need seperate regexes
# test using geodata one for both
## for each item in dataTypes, compile a regex for that item and find all the matches with jsonStr
## and put those matches in dataList.
for dataType in dataTypes:
    p = re.compile("\\" + dataType + "\": ([^,]+)[,|]*)"
    dataList = p.findall(jsonStr)
    for i, val in enumerate(dataList):
        outDict[unicode(str(i))][unicode(dataType)] = convertServerData(val)

## outDict is a multidimensional dict that now holds a val in each dataType per i
## same as above just for geoDataTypes
for geoDataType in geoDataTypes:
    p = re.compile("\\" + geoDataType + "\": (\\"[^\"]*"|"[^"]*"|'[^']*')*)"
    dataList = p.findall(jsonStr)
    for i, val in enumerate(dataList):
        outDict[unicode(str(i))][unicode("geo_data")][unicode(geoDataType)] = convertServerData(val)
## outDict in geo_data for each geoDataType holds a val
return outDict
get_args()

"""
opens up a url to a server, parses server data from there
from that data, creates addresses to each squid, stores in cvmfs_http_proxy
overwrites cvmfs config file with cvmfs_http_proxy if dump is not specified
"""
## CVMFS = CERN Virtual Machine File System

## reads in server data (if it can be read in) into a dictionary called data
try:
    f = urlopen(server)
    # data = json.loads(f.read())
    data = parseServerData(f.read())
except (urllib2.URLError, ValueError), e:
    logging.error("Unable to open url. %s" % e)

```

```

data = None

if data:
    ## iterates through the data dict and uses all hostname and squid_port keys
    ## to create addresses for each squid in closest_http_proxy
    for i in range(0, len(data)):
        try:
            closest_http_proxy += 'http://%s:%s;' % (data['%s%i']['hostname'], data['%s%i']['squid_port'])
            http_proxy_formatted+='http://%s:%s' % (data['%s%i']['hostname'], data['%s%i']['squid_port'])+ "/"
        except KeyError, e:
            logging.error("The data returned from '%s' was missing the key: %s. Please ensure the url is running
the latest Shoal-Server." % (server, e))
            sys.exit(1)

cvmfs_http_proxy += "\"" + closest_http_proxy + "\"\n"

## if dump -> don't overwrite CVMFS config file
if dump:
    #print "%s would have been written to the CVMFS config file", cvmfs_http_proxy
    print "export http_proxy="+http_proxy_formatted

else:
    # attempt to read the cvmfs_config file, no point in continuing if it can't be read
    try:

```

```

        f = open(cvmfs_config)
        lines = f.readlines()
    except:
        logging.error("Could not open and read the CVMFS config file")
        sys.exit(1)

f.close()

# create a list of tuples of lines/line numbers that have "CVMFS_HTTP_PROXY"
CVMFS_proxy_lines = [t for t in enumerate(lines) if "CVMFS_HTTP_PROXY" in t[1]]

# if there is only one can just replace it with the new proxy
if len(CVMFS_proxy_lines) == 1:
    lines[CVMFS_proxy_lines[0][0]] = cvmfs_http_proxy
# add a line if it doesn't exist
elif len(CVMFS_proxy_lines) == 0:
    lines += cvmfs_http_proxy
# something is wrong with the CVMFS config; there are multiple entries
# fixing it by changing the first and removing the extras
else:
    logging.error("CVMFS file had duplicate CVMFS_HTTP_PROXY entries;" +
        " writing the first deleting the rest")
    lines[CVMFS_proxy_lines[0][0]] = cvmfs_http_proxy
    for line in CVMFS_proxy_lines[1:]:
        lines[line[0]] = ""
# open the file again this time for writing and replace its contents with the modified lines

```

```

try:
    try:
        f = open(cvmfs_config, "w")
        f.writelines(lines)
    except Exception:
        logging.error("Could not write CVMFS config file")
finally:
    f.close()

```

```

-----
-----
-----

```

```

File: puppetrepo-shoal/files/shoal-server.py
#!/usr/bin/python
import os
import sys
import logging

from threading import Thread
from time import sleep
from shoal_server import config
from shoal_server.shoal import ThreadMonitor, WebpyServer

def main():

    shoal_list = {}
    threads = []

    # establishes ThreadMonitor and its thread
    monitor_thread = ThreadMonitor(shoal_list)
    monitor_thread.daemon = True
    threads.append(monitor_thread)

    # establishes WebpyServer and its thread
    webpy_thread = WebpyServer(shoal_list)
    webpy_thread.daemon = True
    threads.append(webpy_thread)

    # starts running threads

```

```

for thread in threads:
    thread.start()

# keep running threads until KeyboardInterrupt
#try:
#while True:
#    for thread in threads:
#        if not thread.is_alive():
#            logging.error('{0} died.'.format(thread))
#            sys.exit()
#    #sleep(1)
#except KeyboardInterrupt:
#    sys.exit()
if __name__ == '__main__':
    # sets up logging file
    shoal_dir = config.shoal_dir
    log_file = config.log_file
    log_format = '%(asctime)s - %(levelname)s - [%s:%s] - %s'

    try:
        logging.basicConfig(level=logging.ERROR, format=log_format, filename=log_file)
    except IOError as e:
        sys.exit(1)

    # change working directory so webpy static files load correctly.
    try:
        os.chdir(shoal_dir)
    except OSError as e:
        sys.exit(1)
    main()

```

```

-----
-----
-----

```

```

File: puppetrepo-shoal/files/shoal-server.sh
#!/bin/bash
#This script requires puppet installed.
#One needs to be root to apply this.
#This script install shoal server

if [ "`whoami`" != "root" ]; then
    echo "This should be run by root"
    exit 1
fi

```

```

mkdir -p /etc/puppet/modules

if [ ! -f "/usr/bin/git" ]; then
  yum install git -y
fi

if [ ! -d "/etc/puppet/modules/shoal" ]; then
  git clone ssh://p-puppetrepo@cdcvcs.fnal.gov/cvs/projects/puppetrepo-shoal
  mkdir -p /etc/puppet/modules/shoal && cp -r puppetrepo-shoal/* /etc/puppet/modules/shoal
else
  cd puppetrepo-shoal
  git pull
  cd ..
  cp -rf puppetrepo-shoal/* /etc/puppet/modules/shoal
fi

if [ ! -d "/etc/puppet/modules/rabbitmq" ]; then
  git clone https://github.com/jcochard/puppet-rabbitmq.git
  mkdir -p /etc/puppet/modules/rabbitmq && mv puppet-rabbitmq/* /etc/puppet/modules/rabbitmq
fi

if [ ! -d "/etc/puppet/modules/erlang" ]; then
  puppet module install dcarley/erlang
fi

if [ ! -d "/etc/puppet/modules/epel" ]; then
  puppet module install stahma/epel
fi

if [ ! -d "/etc/puppet/modules/stdlib" ]; then
  puppet module install puppetlabs/stdlib
fi

cat << EOF | puppet apply
include epel
include shoal::server_dependencies
include erlang
include rabbitmq
include shoal::server
include shoal::host_server
Class['epel'] -> Class['shoal::server_dependencies'] -> Class['erlang'] -> Class[rabbitmq] ->
Class['shoal::server'] -> Class['shoal::host_server']
EOF

```

```

-----
-----

```

```

-----

File: puppetrepo-shoal/files/shoal.conf
WSGIDaemonProcess shoal user=apache group=apache threads=10 processes=1
WSGIScriptAlias / /var/www/shoal/scripts/shoal_wsgi.py
WSGIProcessGroup shoal

```

```
Alias /static /var/www/shoal/static/
```

```
AddType text/html .py
```

```
<Directory /var/www/shoal/>
  Order deny,allow
  Allow from all
</Directory>
```

```

-----
-----

```

```

File: puppetrepo-shoal/files/squid.conf
acl NET_LOCAL src 10.0.0.0/8 172.16.0.0/12 192.168.0.0/16
acl HOST_MONITOR src 131.225.152.0/23
acl snmppublic snmp_community HOST_MONITOR
acl all src all
acl manager proto cache_object
acl localhost src 127.0.0.1/32
acl to_localhost dst 127.0.0.0/8 0.0.0.0/32
acl localnet src 10.0.0.0/8 # RFC1918 possible internal network
acl localnet src 172.16.0.0/12 # RFC1918 possible internal network
acl localnet src 192.168.0.0/16 # RFC1918 possible internal network
acl SSL_ports port 443
acl Safe_ports port 80 # http
acl Safe_ports port 21 # ftp
acl Safe_ports port 443 # https
acl Safe_ports port 70 # gopher
acl Safe_ports port 210 # wais
acl Safe_ports port 1025-65535 # unregistered ports
acl Safe_ports port 280 # http-mgmt
acl Safe_ports port 488 # gss-http
acl Safe_ports port 591 # filemaker
acl Safe_ports port 777 # multiling http
acl CONNECT method CONNECT
http_access allow manager localhost
http_access deny manager
http_access deny !Safe_ports

```

```

http_access deny CONNECT !SSL_ports
http_access allow NET_LOCAL
acl our_networks src 131.225.0.0/16 127.0.0.1
http_access allow our_networks
http_access allow localhost
http_access deny all
acl PURGE method PURGE
http_access allow PURGE localhost
http_access deny PURGE
reply_body_max_size 1000000000 allow all
icp_access allow localnet
icp_access deny all
http_port 3128
hierarchy_stoplist cgi-bin
cache_mem 1024 MB
maximum_object_size_in_memory 256 KB
cache_dir ufs /var/spool/squid 290000 16 256
maximum_object_size 10 GB
logformat awstats %>a %ui %un [%d/%b/%Y:%H:%M:%S +0000]t1l "%rm %ru HTTP/%rv" %Hs %<st %Ss:%Sh %tr "%{X-
Frontier-Id}>h" "%{Referer}>h" "%{User-Agent}>h"
access_log /var/log/squid/access.log awstats
logfile_daemon /usr/libexec/squid/logfile-daemon
cache_log /var/log/squid/cache.log
cache_store_log none
mime_table /etc/squid/mime.conf
pid_filename /var/run/squid/squid.pid
strip_query_terms off
unlinkd_program /usr/libexec/squid/unlinkd
refresh_pattern ^ftp:          1440 20% 10080
refresh_pattern ^gopher:      1440 0% 1440
refresh_pattern -i /cgi-bin/   0 0% 0
refresh_pattern \.crl$        60 25% 1440
refresh_pattern \.der$        60 25% 1440
refresh_pattern \.pem$        60 25% 1440
refresh_pattern \.r0$         60 25% 1440
refresh_pattern \.pacman$     60 10% 1440
refresh_pattern .             60 20% 4320
negative_ttl 1 minute
acl shoutcast rep_header X-HTTP09-First-Line ^ICY.[0-9]
upgrade_http0.9 deny shoutcast
acl apache rep_header Server ^Apache
broken_vary_encoding allow apache
collapsed_forwarding on
connect_timeout 30 seconds
read_timeout 1 minute
request_timeout 1 minute
client_lifetime 1 hour
cache_mgr fermigrid-help@fnal.gov

```

```

cache_effective_user squid
cache_effective_group squid
umask 022
snmp_access allow snmppublic HOST_MONITOR
snmp_access deny all
icp_port 0
icon_directory /usr/share/squid/icons
error_directory /usr/share/squid/errors/English
ignore_ims_on_miss on
coredump_dir /var/spool/squid

```

```

-----
-----
-----

```

```

File: puppetrepo-shoal/manifests/agent.pp
# The shoal agent runs in the squid server and announces it to the shoal server

```

```

class shoal::agent(
  $shoal_server_ip = undef,
)
{
  if($shoal_server_ip)
  {
    package { ["shoal-agent"]:
      ensure => installed,
    }

    file_line { ['/etc/shoal/shoal_agent.conf']:
      path => '/etc/shoal/shoal_agent.conf',
      line => "amqp_server_url = ${shoal_server_ip}",
      match => "amqp_server_url =.*$",
      require => Package["shoal-agent"],
      notify => Service["shoal-agent"],
    }

    service { ['shoal-agent']:
      ensure => running,
      enable => true,
      hasstatus => false,
      #hasrestart => true,
      path => "/usr/bin",
      #require => File_line["shoal-agent"],
    }
  }
}

```

```

}
else{
  warning('Shoal agent is NOT INSTALLED. please pass a valid ip to the parameter "shoal_server_ip" and run it
again' )
  warning('eg: shoal_server_url => shoalserver.domain')
}
}
}

```

```

-----
-----
-----

```

File: puppetrepo-shoal/manifests/client.pp

The shoal client runs in the WorkeNodes and sets up the squid proxy according to what the server tells

```

class shoal::client(
  $shoal_server_url = undef,
)
{
  if($shoal_server_url)
  {
    package { 'git':
      ensure => installed,
      before => Exec['git-shoal-client'],
    }

    exec { 'git-shoal-client':
      command => "git clone git://github.com/hep-gc/shoal.git",
      path => "/usr/bin",
      cwd => "/usr",
      require => Package['git'],
      before => Exec['install-shoal-client'],
      creates => "/usr/shoal"
    }

    exec { 'install-shoal-client':
      command => "python setup.py install",
      cwd => "/usr/shoal/shoal-client/",
      path => "/usr/bin",
      require => Exec['git-shoal-client'],
      creates => "/etc/shoal/shoal_client.conf",
    }
  }
}

```

```

file_line { '/etc/shoal/shoal_client.conf':
  path => '/etc/shoal/shoal_client.conf',
  line => "shoal_server_url = ${shoal_server_url}",
  require => Exec['install-shoal-client'],
  match => "shoal_server_url =.*$",
}

file { '/usr/bin/shoal-client':
  ensure => present,
  owner => 'root',
  group => 'root',
  mode => '0755',
  source => "puppet:///modules/shoal/shoal-client",
  require => Exec['install-shoal-client'],
}

exec { 'run-shoal-client':
  command => "shoal-client",
  path => "/usr/bin",
  require => File['/usr/bin/shoal-client'],
  refreshonly => true,
  subscribe => File["/usr/bin/shoal-client"],
}

cron::job{ 'shoal-client':
  minute => '0,30',
  hour => '*',
  date => '*',
  month => '*',
  weekday => '*',
  user => 'root',
  command => '/usr/bin/shoal-client >> /var/log/cron_shoal_client.log',
  environment => [ 'MAILTO=psandeep@hawk.iit.edu' ];
}

file { '/usr/bin/set-session-proxy.sh':
  ensure => present,
  owner => 'root',
  group => 'root',
  mode => '0755',
  source => "puppet:///modules/shoal/set-session-proxy.sh",
}
}

```

```
    else {
      warning('Shoal client is NOT INSTALLED. please pass a valid address to the parameter "shoal_server_url" and
run it again' )
      warning('eg: shoal_server_url => \'http://shoalserver.domain/nearest\' ')
    }
  }
}
#exec { 'run-set-session-proxy-script':
# command => "source set-session-proxy.sh",
# path => "/usr/bin",
# require => File['Add script to set session proxy'],
#}

-----
-----
-----
```

```
File: puppetrepo-shoal/manifests/frontier.pp
# == Class: frontier::squid

#

# Installation and configuration of a frontier squid

#

# === Parameters

#

# [*customize_file*]
# The customization config file to be used.
#

# [*customize_template*]
# The customization config template to be used.
#
```

```
# [*cache_dir*]
# The cache directory.
#

# [*install_resource*]
# The cache directory.
#

# [*resource_path*]
# The cache directory.
#

# === Examples

#

# class { frontier::squid:
#   customize_file => 'puppet:///modules/mymodule/customize.sh',
#   cache_dir      => '/var/squid/cache'
# }

#

# === Authors

#

# Alessandro De Salvo <Alessandro.DeSalvo@romal.infn.it>

#

# === Copyright

#

# Copyright 2014 Alessandro De Salvo

#

# Added comment
```

```
class shoal::frontier (
  $customize_file = undef,
  $customize_template = undef,

  $cache_dir = $shoal::frontier_params::frontier_cache_dir,
  $install_resource = false,
  $resource_path = $shoal::frontier_params::resource_agents_path
) inherits shoal::frontier_params {
  yumrepo {'cern-frontier':
    baseurl => 'http://frontier.cern.ch/dist/rpms/',
    enabled => 1,
    gpgcheck => 1,
    gpgkey => 'http://frontier.cern.ch/dist/rpms/cernFrontierGpgPublicKey'
  }

  package {$shoal::frontier_params::frontier_packages:
    ensure => latest,
    require => Yumrepo['cern-frontier'],
    notify => Service[$shoal::frontier_params::frontier_service]
  }

  if ($cache_dir) {
    file { $cache_dir:
      ensure => directory,
```

```
      owner => squid,
      group => squid,
      mode => 0755,
      require => Package[$shoal::frontier_params::frontier_packages],
      notify => Service[$shoal::frontier_params::frontier_service]
    }
  }

  if ($customize_file) {
    file {$shoal::frontier_params::frontier_customize:
      ensure => file,
      owner => squid,
      group => squid,
      mode => 0555,
      source => $customize_file,
      require => Package[$shoal::frontier_params::frontier_packages],
      notify => Service[$shoal::frontier_params::frontier_service]
    }
  }

  if ($customize_template) {
    file {$shoal::frontier_params::frontier_customize:
      ensure => file,
      owner => squid,
```

```
        group => squid,
        mode  => 0755,
        content => template($customize_template),
        require => Package[$shoal::frontier_params::frontier_packages],
        notify => Service[$shoal::frontier_params::frontier_service]
    }
}

if ($install_resource) {
    file { $resource_path:
        ensure => directory,
        owner  => "root",
        group  => "root",
        mode   => 0755,
    }

    file { "${resource_path}/FrontierSquid":
        ensure => file,
        owner  => "root",
        group  => "root",
        mode   => 0755,
        source => "puppet:///modules/frontier/FrontierSquid",
        require => File[$resource_path]
    }
}
```

```
    }

    service {$shoal::frontier_params::frontier_service:
        ensure => running,
        enable => true,
        hasrestart => true,
        require => Package[$shoal::frontier_params::frontier_packages]
    }

    file { "/var/spool/squid":
        ensure => "directory",
        #owner  => "root",

        #group => "wheel",
        mode   => 766,
    }
}

file { 'Replacing default squid config with new config content':
    ensure => present,
    owner  => 'root',
    group  => 'root',
    mode   => '0755',
    path   => '/etc/squid/squid.conf',
    content => inline_template("acl NET_LOCAL src 10.0.0.0/8 172.16.0.0/12 192.168.0.0/16
```

```
acl HOST_MONITOR src 131.225.152.0/23
acl snmppublic snmp_community HOST_MONITOR
acl all src all
acl manager proto cache_object
acl localhost src 127.0.0.1/32
acl to_localhost dst 127.0.0.0/8 0.0.0.0/32
acl localnet src 10.0.0.0/8 # RFC1918 possible internal network
acl localnet src 172.16.0.0/12 # RFC1918 possible internal network
acl localnet src 192.168.0.0/16 # RFC1918 possible internal network
acl SSL_ports port 443
acl Safe_ports port 80 # http
acl Safe_ports port 21 # ftp
acl Safe_ports port 443 # https
acl Safe_ports port 70 # gopher
acl Safe_ports port 210 # wais
acl Safe_ports port 1025-65535 # unregistered ports
acl Safe_ports port 280 # http-mgmt
acl Safe_ports port 488 # gss-http
acl Safe_ports port 591 # filemaker
acl Safe_ports port 777 # multiling http
acl CONNECT method CONNECT
http_access allow manager localhost
http_access deny manager
http_access deny !Safe_ports
```

```
http_access deny CONNECT !SSL_ports
http_access allow NET_LOCAL
acl our_networks src 131.225.0.0/16 127.0.0.1
http_access allow our_networks
http_access allow localhost
http_access deny all
acl PURGE method PURGE
http_access allow PURGE localhost
http_access deny PURGE
reply_body_max_size 1000000000 allow all
icp_access allow localnet
icp_access deny all
http_port 3128
hierarchy_stoplist cgi-bin
cache_mem 1024 MB
maximum_object_size_in_memory 256 KB
cache_dir ufs /var/spool/squid 290000 16 256
maximum_object_size 10 GB
logformat awstats %>a %ui %un [%{%d/%b/%Y:%H:%M:%S +0000}t1] \"%rm %ru HTTP/%rv\" %Hs %<st %Ss:%Sh %tr \"%{X-
Frontier-Id}>h\" \"%{Referer}>h\" \"%{User-Agent}>h\"
access_log /var/log/squid/access.log awstats
logfile_daemon /usr/libexec/squid/logfile-daemon
cache_log /var/log/squid/cache.log
cache_store_log none
```

```

mime_table /etc/squid/mime.conf
pid_filename /var/run/squid/squid.pid
strip_query_terms off
unlinkd_program /usr/libexec/squid/unlinkd
refresh_pattern ^ftp:      1440  20%  10080
refresh_pattern ^gopher:   1440  0%   1440
refresh_pattern -i /cgi-bin/ 0     0%   0
refresh_pattern \\.crl$    60    25%  1440
refresh_pattern \\.der$    60    25%  1440
refresh_pattern \\.pem$    60    25%  1440
refresh_pattern \\.r0$     60    25%  1440
refresh_pattern \\.pacman$ 60    10%  1440
refresh_pattern .         60    20%  4320
negative_ttl 1 minute
acl shoutcast rep_header X-HTTP09-First-Line ^ICY.[0-9]
upgrade_http0.9 deny shoutcast
acl apache rep_header Server ^Apache
broken_vary_encoding allow apache
collapsed_forwarding on
connect_timeout 30 seconds
read_timeout 1 minute

request_timeout 1 minute
client_lifetime 1 hour
cache_mgr fermigrid-help@fnal.gov

```

```

cache_effective_user squid
cache_effective_group squid
umask 022

snmp_access allow snmppublic HOST_MONITOR
snmp_access deny all
icp_port 0
icon_directory /usr/share/squid/icons
error_directory /usr/share/squid/errors/English
ignore_ims_on_miss on
coredump_dir /var/spool/squid
"),
notify => Service[$shoal::frontier_params::frontier_service],
}

}

```

```

-----
-----
-----

```

```

File: puppetrepo-shoal/manifests/frontier_params.pp
#contains parameters for shoal::frontier class
class shoal::frontier_params {
  case $::osfamily {
    'RedHat': {
      $frontier_release_provider = 'rpm'
      $frontier_release_package = 'frontier-release-1.0-1.noarch.rpm'
      $frontier_release_package_url = "http://frontier.cern.ch/dist/rpms/RPMS/noarch/${frontier_release_package}"
      $frontier_packages = ['frontier-squid']
      $frontier_service = 'frontier-squid'
    }
  }
}

```

```

    $frontier_customize = '/etc/squid/customize.sh'
    $frontier_cache_dir = '/var/cache/squid'
    $resource_agents_path = '/usr/lib/ocf/resource.d/lcg'
  }
  default: {
  }
}
}
-----
-----
-----

```

```

File: puppetrepo-shoal/manifests/host_server.pp
# Installs wsgi module for apache and hosts shoal server
class shoal::host_server{

```

```

  $rabbitmq_server_url = "localhost"

  file { ["/var/run/wsgi":
    ensure => "directory",
  ]

  package { [httpd:
    ensure => installed,
  ]

  exec { ['install-wsgi':
    command => "yum install mod_wsgi -y",
    path => "/usr/bin",
    creates => "/usr/lib64/httpd/modules/mod_wsgi.so",
  ]

  file_line { ['Add wsgi module to /etc/httpd/conf/httpd.conf':
    path => '/etc/httpd/conf/httpd.conf',
    line => 'LoadModule wsgi_module modules/mod_wsgi.so',
    require => Exec['install-wsgi'],
    notify => Service["httpd"],
  ]

  file_line { ['/etc/httpd/conf/httpd.conf':
    path => '/etc/httpd/conf/httpd.conf',
    line => 'WSGISocketPrefix /var/run/wsgi',
    require => Exec['install-wsgi'],
    notify => Service["httpd"],
  ]
}

```

```

exec { ['Host shoal on apache':
  command => "mv /var/shoal/ /var/www/",
  path => "/bin",
  creates => "/var/www/shoal",
]

file_line { ['changing shoal dir in /etc/shoal/shoal_server.conf':
  path => '/etc/shoal/shoal_server.conf',
  line => 'shoal_dir = /var/www/shoal/',
  require => Exec['install-wsgi'],
  match => "shoal_dir =.*$",
]

file_line { ['changing amqp server url in /etc/shoal/shoal_server.conf':
  path => '/etc/shoal/shoal_server.conf',
  line => "amqp_server_url = ${rabbitmq_server_url}",
  require => Exec['install-wsgi'],
  match => "amqp_server_url =.*$",
  notify => Service["httpd"],
]

file { ['/etc/httpd/conf.d/shoal.conf':
  ensure => present,
  source => "puppet:///modules/shoal/shoal.conf",
]

file { ["/var/log/shoal_server.log":
  mode => '777',
  #recurse => true,
]

file { ["/var/www/shoal/scripts/shoal_wsgi.py":
  mode => '755',
  #recurse => true,
]

service { ['httpd':
  ensure => running,
  path => "/usr/sbin/",
  require => Exec['install-wsgi'],
]
}
}

```

```
-----  
-----  
-----
```

```
File: puppetrepo-shoal/manifests/replace_squid_conf.pp  
class shoal::replace_squid_conf{
```

```
  file { ["/var/spool/squid":  
    ensure => "directory",  
    mode   => 766,  
  ] }  
  
  file { ["/etc/squid/squid.conf":  
    ensure => present,  
    owner  => 'root',  
    group  => 'root',  
    mode   => '0755',  
    source => "puppet:///modules/shoal/squid.conf",  
    #notify => Service['frontier-squid'],  
    require => File['/var/spool/squid'],  
  ] }  
  
  exec { ['restart frontier-squid':  
    command => "service frontier-squid restart",  
    require => File['/etc/squid/squid.conf'],  
    path    => '/sbin'  
  ] }  
}
```

```
-----  
-----  
-----
```

```
File: puppetrepo-shoal/manifests/repository.pp  
# shoal repository form where we install the shoal agent package
```

```
class shoal::repository {  
  package { ["yum-conf-epel":  
    ensure => installed,  
  ] }  
}
```

```
# exec { ['yum-update':  
  # command => "yum update -y",  
  # path => "/usr/bin",  
  # before => Exec["curl-shoal-repo"],  
  # require => Package["yum-conf-epel"],  
  # creates => "/usr/bin/shoal-agent",  
#}]  
  
  exec { ['curl-shoal-repo':  
    command => "curl http://shoal.heprc.uvic.ca/repo/shoal.repo -o /etc/yum.repos.d/shoal.repo",  
    path => "/usr/bin",  
    before => Exec["rpm-import"],  
    # require => Exec["yum-update"],  
    require => Package["yum-conf-epel"],  
    creates => "/etc/yum.repos.d/shoal.repo",  
  ] }  
  
  exec { ['rpm-import':  
    command => "rpm --import http://hepnetcanada.ca/pubkeys/igable.asc",  
    path => "/bin",  
    require => Exec["curl-shoal-repo"],  
    creates => "/usr/bin/shoal-agent",  
  ] }  
}
```

```
-----  
-----  
-----
```

```
File: puppetrepo-shoal/manifests/server.pp
```

```
# Installs the shoal server  
class shoal::server{  
  
  exec { ['git-shoal-server':  
    command => "git clone https://github.com/SandeepPalur/shoal.git",  
    cwd => "/usr",  
    path => "/usr/bin",  
    before => Exec["install-shoal-server"],  
    creates => "/usr/shoal",  
  ] }  
  
  exec { ['install-shoal-server':  
    command => "python setup.py install",  
    cwd => "/usr/shoal/shoal-server",  
    path => "/usr/bin",  
  ] }  
}
```

```

require => Exec["git-shoal-server"],
creates => "/etc/shoal/shoal_server.conf",
}

exec { "wget-pip":
  command => "wget https://bootstrap.pypa.io/get-pip.py --no-check-certificate",
  before => Exec["install-pip"],
  require => Exec["install-shoal-server"],
  path => "/usr/bin",
  cwd => "/usr",
  creates => "/usr/bin/pip",
}

exec { "install-pip":
  path => "/usr/bin",
  cwd => "/usr",
  command => "python get-pip.py",
  require => Exec["wget-pip"],
  before => Exec["install-webpy"],
  creates => "/usr/bin/pip",
}

exec { "install-webpy":
  path => "/usr/bin",
  command => "pip install web.py",
  require => Exec["install-pip"],
  before => Exec["install-pygeoip"],
  creates => "/usr/lib/python2.6/site-packages/web.py-0.37-py2.6.egg-info",
}

exec { "install-pygeoip":
  path => "/usr/bin",
  command => "pip install pygeoip",
  require => Exec["install-webpy"],
  before => Exec["install-pika"],
  creates => "/usr/lib/python2.6/site-packages/pygeoip",
}

exec { "install-pika":
  path => "/usr/bin",
  command => "pip install pika",
  creates => "/usr/lib/python2.6/site-packages/pika",
}

file { '/usr/bin/shoal-server.py':
  require => Exec["install-pika"],
  ensure => present,
}

```

```

owner => 'root',
source => "puppet:///modules/shoal/shoal-server.py",
}

exec { "Run shoal server script":
  path => "/usr/bin",
  command => "python /usr/bin/shoal-server.py",
  require => File['/usr/bin/shoal-server.py'],
  refreshonly => true,
  subscribe => File["/usr/bin/shoal-server.py"],
}
}

```

```

-----
-----
-----

```

```

File: puppetrepo-shoal/manifests/server_dependencies.pp
# Installs shoal server dependency packages
class shoal::server_dependencies{
  Package { ensure => "installed" }

```

```

  package { "wget": }
  package { "gcc": }
  package { "make": }
  package { "ncurses":}
  package { "ncurses-devel":}
  package { "openssl-devel":}
  package { "libxslt":}
  package { "zip":}
  package { "unzip":}
  package { "nc":}
  package { "git":}

```

```

}

```

```

-----
-----
-----

```